CAPTCHA Solver

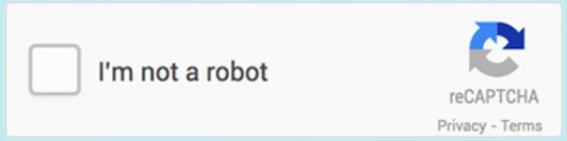
RESOLVE CAPTCHAS WITH COMPUTER VISION
AND CONVOLUTIONAL NEURAL NETWORK

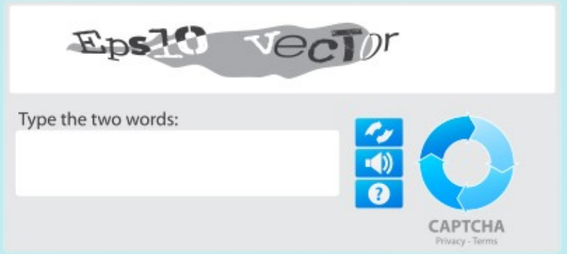
Mariotti – Pizzari – Realacci – Romito

What's a CAPTCHA?

Completely Automated Public Turing test to tell Computers and Humans Apart

- Is a type of virtual challenge
- Aims to determine if a user is a robot or not
- Various types exist

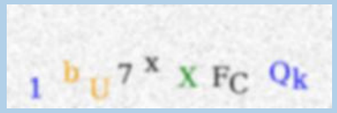


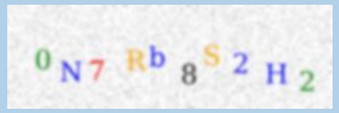


We used two different datasets:

DATASET A:

- Distorted immages
- Line passing through characters
- Grayscale





DATASET B:

- Colored characters
- Placed in random places.
- Lowercase and uppercase

Our approach:

- First idea: “clean a little” our images and train the neural network directly with those. In this way we would have lost a lot of potential.

- Final idea: train the network using individual characters extracted from the CAPTCHAs using OpenCV. This means create a bigger dataset that very often translates to a much more precise guess.

Our approach:

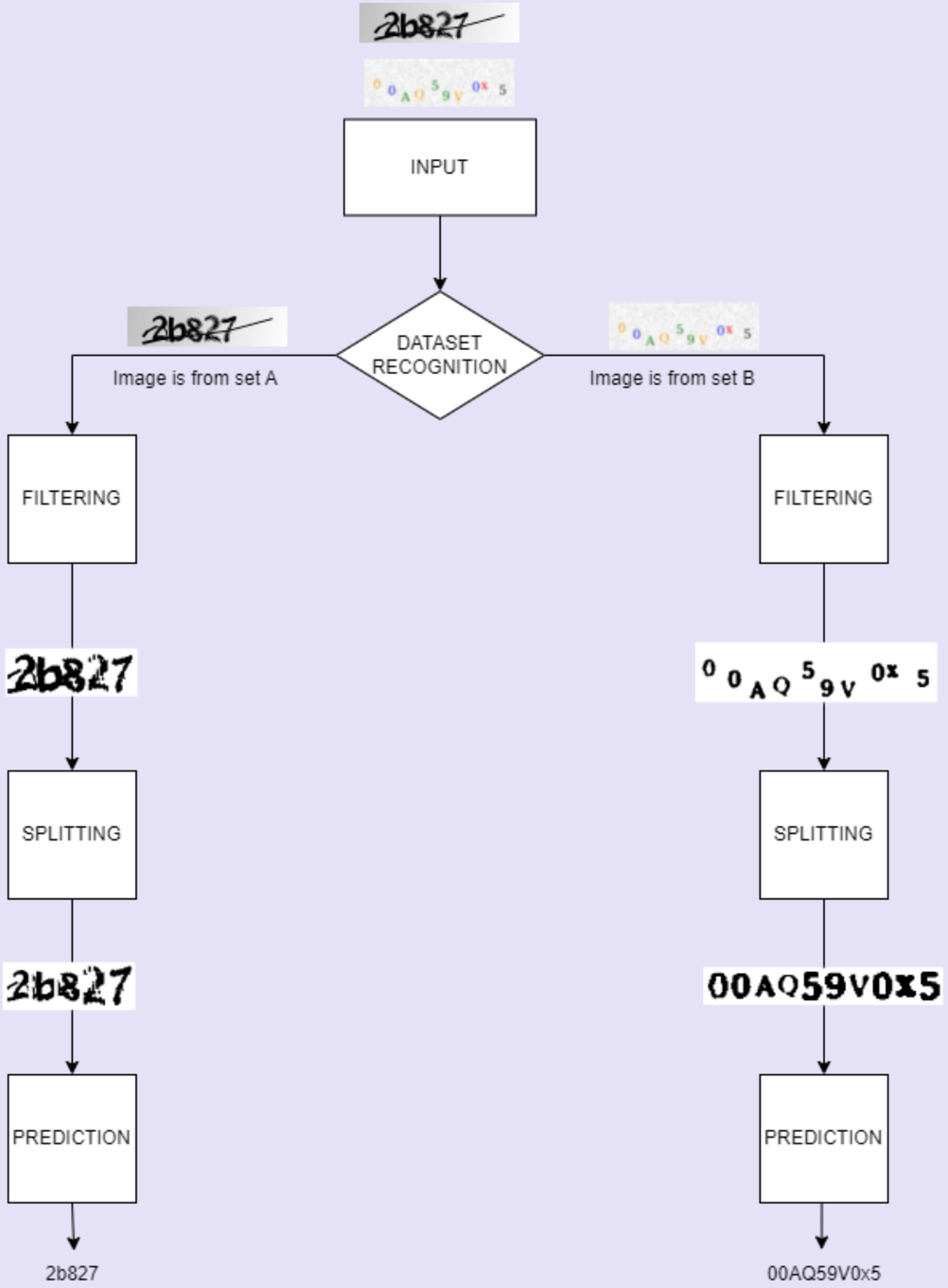
- First idea: “clean a little” our images and train the neural network directly with those. In this way we would have lost a lot of potential.

- Final idea: train the network using individual characters extracted from the CAPTCHAs using OpenCV. This means create a bigger dataset that very often translates to a much more precise guess.

WE USED TWO DIFFERENT MODELS:

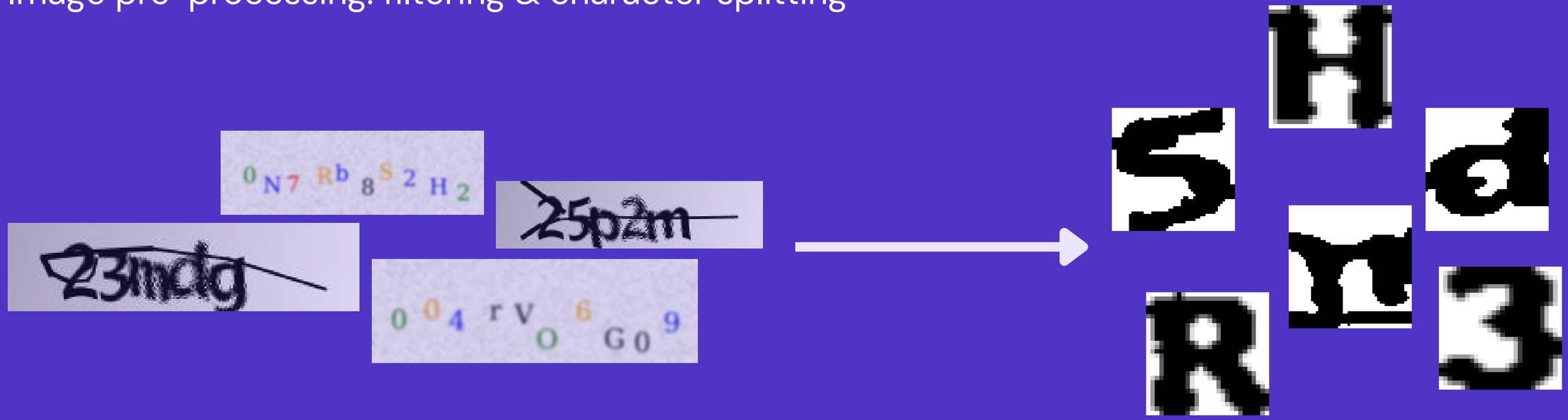
There are a lot of differences between the images, so we can train each model specifically for the type of character!

THE FINAL ARCHITECTURE



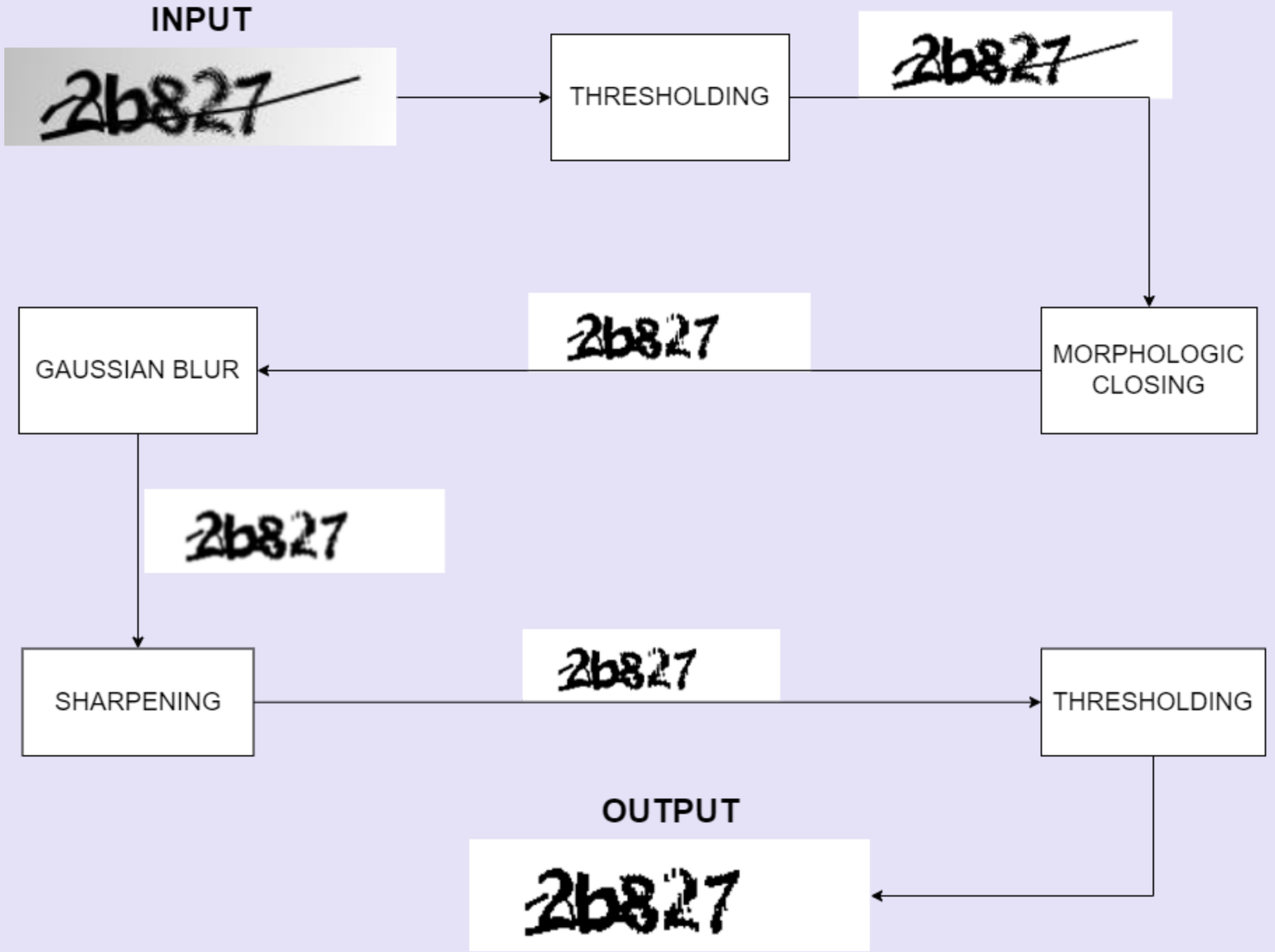
BUILDING THE DATASETS

Image pre-processing: filtering & character splitting



FILTERING

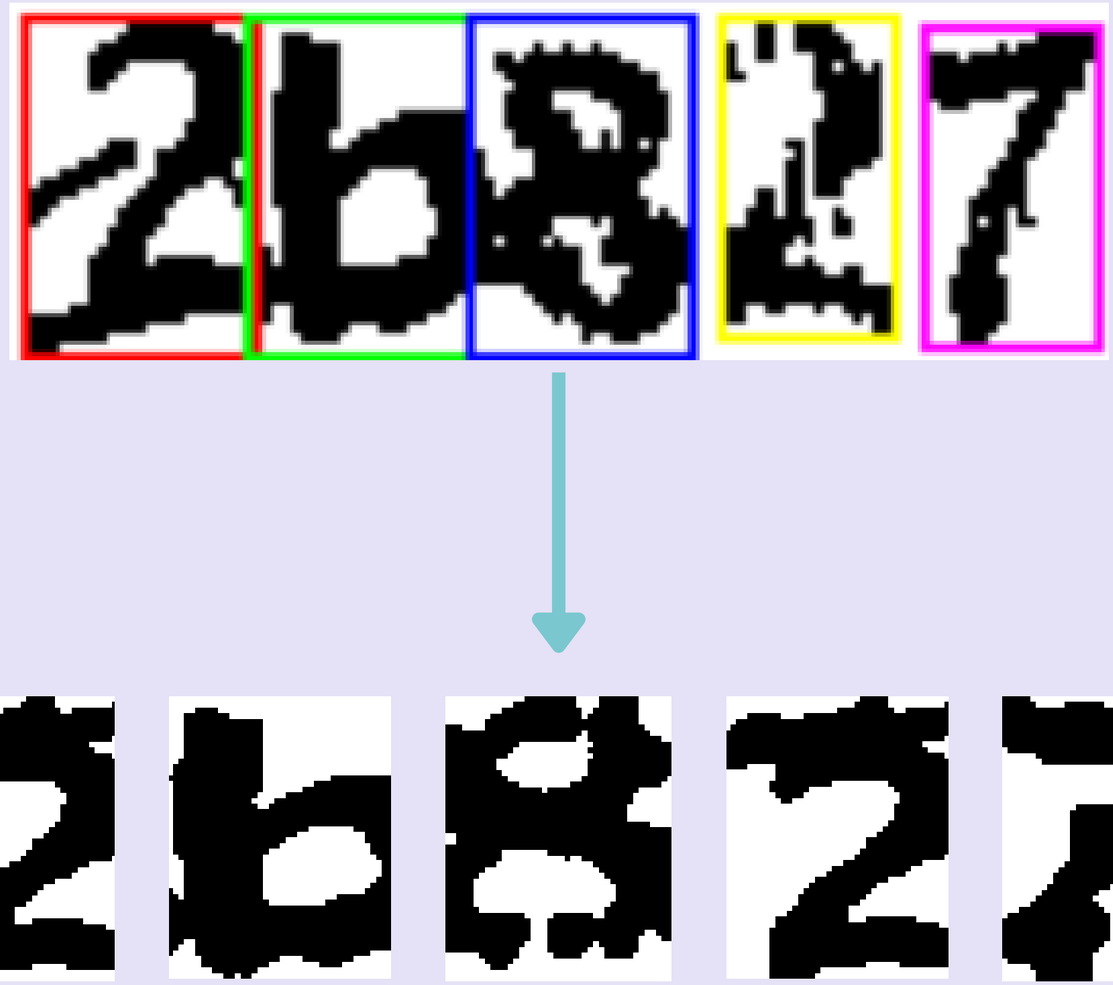
- Thresholding
- Morphologic closing
- Gaussian blur
- Sharpening
- Thresholding



CHARACTER SPLITTING

MSER algorithm to find regions

- Too large region → Split
- Almost all white pixel → Discard
- Region above another → Merge
- Region overlap → Merge
- Too small region → Discard



DATA AUGMENTATION

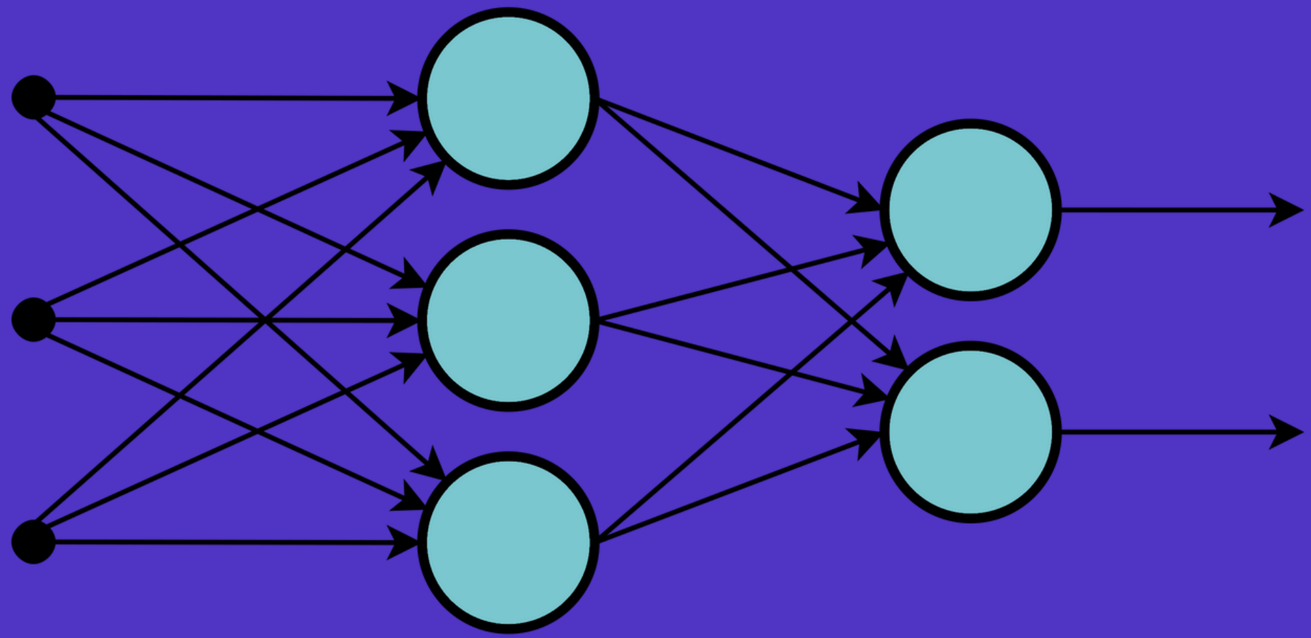
- This dataset was not so large
- Rotate each character by a random angle between -45° and 45°
- Save the image
- Doubled the dataset size

SAVING THE DATA

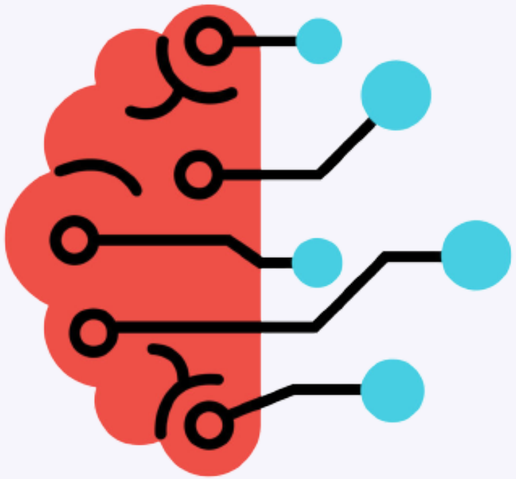
- Image stored for each character with an incremental number as filename
- Different directory for each dataset
- CSV file to associate filename to label
- Label = character represented in the image

DEFINING THE MODELS

For the two types of CAPTCHAs



Model(s) definition



How we defined the models

CONVOLUTIONAL LAYERS

In the CNN we tried different composition of convolutional layers to extract features from the images and we converged on three different layers which resulted in the best performances we could get. We also played around with kernel sizes and strides to impove the model.

NORMALIZATON

We also found out that adding a normalization layer after each convolution helped the network in both performances and stability.

LINEAR LAYERS

After the feature extracting process, which was done in the convolutional layers, we proceeded to linearize the data and we added three linear layers to classify the images.

TRAINING & USING THE MODELS

To solve CAPTCHAs

Training the models

SPLIT THE DATASETS

80/20 split ratio

We split the datasets in a **training set** (80% of the images) and a **testing set** (20% of the images).

TRAINING

30 epoch with batch size of 32

We trained the neural network for 30 epoch each one with 32 CAPTCHA character as input.

VALIDATION

At the end of each epoch

To have confirmation that the training process was proceeding as expected we collected the training loss.

TEST RESULTS

Model A accuracy: 88%

Model B accuracy: 97%

USING THE MODELS

Execution example

File: samples/type_A/728n8.png

Execution result:

```
Processing file: samples/type_A/728n8.png
image is from SET 1
splitting the image in characters...
loading the model...
predicting (using cuda platform)...
Predicted value: 728n8
 ~/a/captcha-solver on main ×
```

Thanks for your attention!